MEA 712: (An Introduction to) Mesoscale Atmospheric Modeling
Midterm "take home" exam, fall 2009
Due: Start of class on Tuesday 20 October

This "exam" assignment is "open book", but you are expected to work on your own.

In coding CMM, we have elected to use the leapfrog scheme (centered in space and time). We will now consider the performance of the leapfrog advection scheme for a variety of conditions. This is meant to build on (i.e. use the same logic as) what you have done in the fifth mini computing assignment.

Again, we will consider the following equation for constant advection:

$$\frac{\partial \psi}{\partial t} = -c\frac{\partial \psi}{\partial x}. \tag{1}$$

Code up the leapfrog finite difference approximation to this equation. We will again vary the initial condition and the current speed, as explained below. You should continue to use a periodic lateral boundary condition (note that its formulation in the leapfrog scheme will need to be slightly modified since you need values on both the inflow and outflow edges of the domain).

1. Perform the following 16 model realizations. Keep Δt and Δx fixed to their original settings from the prior assignments (10.0 s and 100.0 m, respectively). *You should run each realization forward for the number of timesteps that would be required for the* ***physical solution*** *to move eastward by exactly 1500.0 m (please note, this is a different instruction from the original upstream assignment!).*

   a) Start the model with each of the following initial conditions: a cosine wave with wavelengths of 2Δx, 4Δx, 10Δx, and 20Δx. In other words, your initial condition should be

$$\psi(t=0) = \cos\left(\frac{2\pi x}{n\Delta x}\right), \tag{2}$$

   where n=2, 4, 10, and 20.

   b) For *each* wavelength, run the model with varying choices of c such that you are able to test the following parameter space: μ=0.1, 0.5, 1.0, and 1.5, where

$$\mu \equiv c\frac{\Delta t}{\Delta x}. \tag{3}$$

   c) Carefully analyze your results from the 16 model runs. You should inspect *each* run's output at *all* output times to be sure you are seeing the whole picture.

2. Comment on the **speed preservation** of the leapfrog scheme. *Present a table of c_{lf}/c_{phys} values to support your discussion.*

   a) In what way (if at all) is speed preservation a function of the wavelength of the feature?

   b) In what way (if at all) is speed preservation a function of the Courant number, μ?

   c) Considering the logic of the numerical scheme, interpret these properties you have discovered in a and b. Why does the leapfrog scheme behave this way?

(Continued)

3. Comment on the **amplitude preservation** of the leapfrog scheme. *Present a table of λ values to support your discussion.*

a) In what way (if at all) is amplitude preservation a function of the wavelength of the feature?

b) In what way (if at all) is amplitude preservation a function of the Courant number, μ?

c) Considering the logic of the numerical scheme, interpret these properties you have discovered in a and b. Why does the leapfrog scheme behave this way?

4. Compare and contrast the performance of the upstream and leapfrog schemes. In what ways is one superior/inferior to the other? *You may wish to rerun your upstream scheme with the slightly different guidance provided here, in order to make a robust comparison.*

5. Turn in for credit:

a) One plot each of the final time from the following 4 runs: 2Δx wave with μ=1.0; 4Δx wave with μ=0.1; 10Δx wave with μ=0.5; 20Δx wave with μ=1.5. If any of your solutions go to `NAN`, then show the time immediately before all values are `NAN` (i.e. show the final time with finite values, however ugly!). **Be sure each plot is labelled appropriately (including the run name and the timestep at which the plot is valid).**

b) Your written responses to questions 2–4, including tables.

c) Your leapfrog FORTRAN code.

Helpful hints:

- To determine amplitude changes precisely, use a `MAX` function in your FORTRAN code.
- It can be somewhat inaccruate to compute λ after only one timestep. A better idea is to compute it after a number of timesteps to smooth out some of the temporal variability. Just remember that λ is applied ("compounded") on every timestep, so that the total damping over time is λ^n.
- To determine phase speed changes precisely, use a Hovmoller diagram so that you can unambiguously track the waves across the timesteps.